Package 'MSclassifR'

February 25, 2022

Type Package

Title Automated Classification of Mass Spectra

Version 0.2.0

Maintainer Alexandre Godmer <alexandre.godmer@aphp.fr>

Description
Functions to classify mass spectra in known categories, and to determine discriminant mass-over-charge values. It includes easy-to-use functions for pre-processing mass spectra, functions to determine discriminant mass-over-charge values (m/z) from a library of mass spectra corresponding to different categories, and functions to predict the category (species, phenotypes, etc.) associated to a mass spectrum from a list of selected mass-over-charge values. Two vignettes illustrating how to use the functions of this package from real data sets are also available online to help users: <https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Ecrobia.html> and <https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Klebsiella.html>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 4.0)

Imports e1071, MALDIquant, MALDIrppa, MALDIquantForeign, mixOmics, caret, reshape2, ggplot2, nnet, dplyr, fuzzyjoin, VSURF, metap, xgboost, glmnet,

Suggests knitr, rmarkdown,

NeedsCompilation yes

RoxygenNote 7.1.1

Author Alexandre Godmer [aut, cre], Quentin Giai Gianetto [aut]

Repository CRAN

Date/Publication 2022-02-25 12:20:07 UTC

R topics documented:

`CitrobacterRKImetadata` *Metadata of mass spectra corresponding to the bacterial species* Citrobacter *sp. from The Robert Koch-Institute (RKI) database of microbial MALDI-TOF mass spectra*

Description

Metadada of the `CitrobacterRKIspectra` list of mass spectra.

Usage

```
data("CitrobacterRKImetadata", package = "MSclassifR")
```

Format

A data frame with 14 rows (each corresponding to a mass spectrum), and five columns that contain (in order): the strain name, the species name, the spot, a sample number and the name of the strain associated with the spot.

Details

The Robert Koch-Institute (RKI) database of microbial MALDI-TOF mass spectra contains raw mass spectra. Only mass spectra of the *Citrobacter* bacterial species were collected. Metadata were manually reported from raw data.

Source

The raw data were downloaded from this link : https://zenodo.org/record/163517#.YIkWiNZuJCp. The dataset focuses only on mass spectra from *Citrobacter*.

References

Lasch, Peter, Stammler, Maren, & Schneider, Andy. (2018). Version 3 (20181130) of the MALDI-TOF Mass Spectrometry Database for Identification and Classification of Highly Pathogenic Microorganisms from the Robert Koch-Institute (RKI) [Data set]. Zenodo.doi: 10.5281/zenodo.163517

`CitrobacterRKIspectra` *Mass spectra corresponding to the bacterial species* Citrobacter *sp. from The Robert Koch-Institute (RKI) database of microbial MALDI-TOF mass spectra*

Description

Mass spectra of the `CitrobacterRKIspectra` dataset.

Usage

```
data("CitrobacterRKIspectra", package = "MSclassifR")

#####
#Plotting the first mass spectrum
#library("MSclassifR")
#PlotSpectra(SpectralData=CitrobacterRKIspectra[[1]],absx = "ALL", Peaks = NULL,
#            Peaks2 = NULL, col_spec = 1, col_peak = 2, shape_peak = 3,
#            col_peak2 = 2, shape_peak2 = 2)
```

Format

A list that contains 14 objects of class S4 corresponding each to a each mass spectrum.

Details

The Robert Koch-Institute (RKI) database of microbial MALDI-TOF mass spectra contains raw mass spectra. Only mass spectra of the *Citrobacter* bacterial species were collected.

Source

The raw data were downloaded from this link : `https://zenodo.org/record/163517#.YIkWiNZuJCp`. The dataset focuses only on mass spectra from *Citrobacter*.

References

Lasch, Peter, Stammler, Maren, & Schneider, Andy. (2018). Version 3 (20181130) of the MALDI-TOF Mass Spectrometry Database for Identification and Classification of Highly Pathogenic Microorganisms from the Robert Koch-Institute (RKI) [Data set]. Zenodo.doi: 10.5281/zenodo.163517

LogReg *Estimation of a multinomial logistic regression to predict the category to which a mass spectrum belongs*

Description

This function estimates a multinomial logistic regression using cross-validation to predict the category (species, phenotypes...) to which a mass spectrum belongs from a set of shortlisted mass-over-charge values corresponding to discriminant peaks. Two main kinds of models can be estimated: linear or nonlinear (with neural networks, random forests, support vector machines with linear kernel, or eXtreme Gradient Boosting). Hyperparameters are randomly searched, except for the eXtreme Gradient Boosting where a grid search is performed.

Usage

```
LogReg(X, moz, Y, number = 2, repeats = 2, kind="linear")
```

Arguments

`X`	`matrix` corresponding to a library of mass spectra. Each row of `X` is the intensities of a mass spectrum measured on the `moz` values.
`moz`	`vector` with shortlisted mass-over-charge values.
`Y`	`factor` with a length equal to the number of rows in `X` and containing the categories of each mass spectrum in `X`.
`number`	`integer` corresponding to the number of folds or number of resampling iterations. See arguments of the `trainControl` function of the `caret` R package.
`repeats`	`integer` corresponding to the number of complete sets of folds to compute. See `trainControl` function of the `caret` R package for more details.
`kind`	If `kind="nnet"`, then a nonlinear multinomial logistic regression is estimated via neural networks. If `kind="rf"`, then it is estimated via random forests. If `kind="svm"`, then it is estimated via support vector machines with linear kernel. If `kind="xgb"`, then it is estimated via eXtreme gradient boosting. Else a linear multinomial logistic regression is performed (by default).

Details

This function estimates a model from a library of mass spectra for which we already know the category to which they belong (ex.: species, etc). This model can next be used to predict the category of a new coming spectrum for which the category is unknown (see `Predict_LogReg`). The estimation is performed using the `train` function of the `caret` R package. For each kind of model, random parameters are tested to find a model with the best Accuracy.

Value

Returns a `list` with four items:

`train_mod`	a `list` corresponding to the output of the train function of the `caret` R package containing the multinomial regression model estimated using repeated cross-validation.
`conf_mat`	a confusion matrix containing percentages classes of predicted categories in function of actual categories, resulting from repeated cross-validation.
`stats_global`	a `data frame` containing the mean and standard deviation values of the "Accuracy"" and "Kappa" parameters computed for each cross-validation.
`boxplot`	a `ggplot` object (see `ggplot2` R package). This is a graphical representation of the "Accuracy" and "Kappa"" parameters of `stats_global` using boxplots.

References

Kuhn, M. (2008). Building predictive models in R using the caret package. Journal of statistical software, 28(1), 1-26.

Examples

```
library("MSclassifR")
library("MALDIquant")
library("mixOmics")

####################################
## 1. Pre-processing of mass spectra

# load mass spectra and their metadata
data("CitrobacterRKIspectra","CitrobacterRKImetadata", package = "MSclassifR")
# standard pre-processing of mass spectra
spectra <- SignalProcessing(CitrobacterRKIspectra)
# detection of peaks in pre-processed mass spectra
peaks <- PeakDetection(x = spectra, labels = CitrobacterRKImetadata$Strain_name_spot)
# matrix with intensities of peaks arranged in rows (each column is a mass-over-charge value)
IntMat <- MALDIquant::intensityMatrix(peaks)
rownames(IntMat) <- paste(CitrobacterRKImetadata$Strain_name_spot)
# remove missing values in the matrix
IntMat[is.na(IntMat)] <- 0
# normalize peaks according to the maximum intensity value for each mass spectrum
IntMat <- apply(IntMat,1,function(x) x/(max(x)))
# transpose the matrix for statistical analysis
X <- t(IntMat)
# define the known categories of mass spectra for the classification
Y <- factor(CitrobacterRKImetadata$Species)

#####################################################################
## 2. Selection of discriminant mass-over-charge values using sPLS-DA
```

```
a <- SelectionVar(X,
                  Y,
                  MethodSelection = c("RFERF"),
                  MethodValidation = c("cv"),
                  PreProcessing = c("center","scale","nzv","corr"),
                  NumberCV = 2,
                  Sizes = c(5:10))
sel_moz=a$sel_moz

#########################################################################
## 3. Perform LogReg from shortlisted discriminant mass-over-charge values

#linear multinomial regression
model_lm=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2)
#Estimated model:
model_lm

#nonlinear multinomial regression using neural networks
model_nn=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="nnet")
#Estimated model:
model_nn

#nonlinear multinomial regression using random forests
model_rf=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="rf")
#Estimated model:
model_rf

#nonlinear multinomial regression using xgboost
model_xgb=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="xgb")
#Estimated model:
model_xgb

#nonlinear multinomial regression using svm
model_svm=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="svm")
#Estimated model:
model_svm

##########
#Of note, step 3 can be performed several times to find optimal models
#because of random hyperparameter search

#####################################
## 4. Select best models in term of average accuracy and saving it for reuse

acc_model=c(model_lm$stats_global[1,1],model_nn$stats_global[1,1],
model_rf$stats_global[1,1],model_xgb$stats_global[1,1],model_svm$stats_global[1,1])
names(acc_model)=c("lm","nn","rf","xgb","svm")
#Best models in term of accuracy
acc_model[which(acc_model==max(acc_model))]

#save best models for reuse
```

```
#models=list(model_lm$train_mod,model_nn$train_mod,model_rf$train_mod,
#model_xgb$train_mod,model_svm$train_mod)
#models_best=models[which(acc_model==max(acc_model))]
#for (i in 1:length(models_best)){
#save(models_best[[i]], file = paste0("model_best_",i,".rda",collapse="")
#}

#load a saved model
#load("model_best_1.rda")
```

`MSclassifR` *Automated classification of mass spectra*

Description

This package provides R functions to classify mass spectra in known categories, and to determine discriminant mass-over-charge values. It was developed with the aim of identifying very similar species or phenotypes of bacteria from mass spectra obtained by Matrix Assisted Laser Desorption Ionisation - Time Of Flight Mass Spectrometry (MALDI-TOF MS). However, the different functions of this package can also be used to classify other categories associated to mass spectra; or from mass spectra obtained with other mass spectrometry techniques. It includes easy-to-use functions for pre-processing mass spectra, functions to determine discriminant mass-over-charge values (m/z) from a library of mass spectra corresponding to different categories, and functions to predict the category (species, phenotypes, etc.) associated to a mass spectrum from a list of selected mass-over-charge values.

Value

No return value. Package description.

Author(s)

Alexandre Godmer, Quentin Giai Gianetto

`PeakDetection` *Detection of peaks in* `MassSpectrum` *objects.*

Description

This function performs a data analysis pipeline to pre-process mass spectra. It provides average intensities and detects peaks using functions of R packages `MALDIquant` and `MALDIrppa`.

Usage

```
PeakDetection(x,
              labels,
              averageMassSpectraMethod = "median",
              SNRdetection = 3,
              halfWindowSizeDetection = 11,
              AlignFrequency = 0.20,
              AlignMethod = "strict",
              Tolerance = 0.002,
              ...)
```

Arguments

x	a list of MassSpectrum objects (see MALDIquant R package).
labels	a list of factor objects to do groupwise averaging.
averageMassSpectraMethod	a character indicating the method used to average mass spectra according to labels. It is fixed to "median" by default. See averageMassSpectra of MALDIquant R package.
SNRdetection	a numeric value indicating the signal-to-noise ratio used to detect peaks (by default = 3). See detectPeaks-methods of the MALDIquant R package for details.
halfWindowSizeDetection	a numeric value half window size to detect peaks (by default = 11). See detectPeaks-methods of the MALDIquant R package for details.
AlignFrequency	a numeric value used to align and bin mass spectra using minimum relative frequency. See alignPeaks of the MALDIrppa R package for more details.
AlignMethod	a character indicating the method used to equalize masses for similar peaks. The strict method is used by default. See binPeaks of the MALDIquant R package for more details.
Tolerance	a numeric value corresponding to the maximal deviation in peak masses to be considered as identical (by default = 0.002). See determineWarpingFunctions of the MALDIquant R package for details.
...	other arguments from MALDIquant and MALDIrppa packages.

Value

Returns a list of MassPeaks objects (see MALDIquant R package) for each mass spectrum in x.

References

Gibb S, Strimmer K. MALDIquant: a versatile R package for the analysis of mass spectrometry data. Bioinformatics. 2012 Sep 1;28(17):2270-1. doi: 10.1093/bioinformatics/bts447. Epub 2012 Jul 12. PMID: 22796955.

Javier Palarea-Albaladejo, Kevin Mclean, Frank Wright, David G E Smith, MALDIrppa: quality control and robust analysis for mass spectrometry data, Bioinformatics, Volume 34, Issue 3, 01 February 2018, Pages 522 - 523, doi: 10.1093/bioinformatics/btx628

See Also

Vignettes MSclassifR : https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Ecrobia.html https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Klebsiella.html

Examples

```
library("MALDIquant")
library("MALDIquantForeign")
library("MSclassifR")


## Load mass spectra and metadata
data("CitrobacterRKIspectra", "CitrobacterRKImetadata", package = "MSclassifR")

## Pre-processing of mass spectra
spectra <- SignalProcessing(CitrobacterRKIspectra)

## Detection of peaks in pre-processed mass spectra
peaks <- PeakDetection(x = spectra,
 labels = CitrobacterRKImetadata$Strain_name_spot,
 averageMassSpectraMethod = "median",
 SNRdetection = 3,
 halfWindowSizeDetection = 11,
 AlignFrequency = 0.20,
 AlignMethod = "strict",
 Tolerance = 0.002)

# Plot peaks on a pre-processed mass spectrum
PlotSpectra(SpectralData=spectra[[1]],Peaks=peaks[[1]],col_spec="blue",col_peak="black")
```

`PlotSpectra` *Plot mass spectra with detected peaks*

Description

This function performs a plot of a `AbstractMassObject` object (see the `MALDIquant` R package). It can be used to highlight peaks in a mass spectrum.

Usage

```
PlotSpectra(SpectralData, absx="ALL", Peaks=NULL, Peaks2=NULL, col_spec=1,
            col_peak=2, shape_peak=3, col_peak2=2, shape_peak2=2)
```

Arguments

SpectralData	MassSpectrum object of S4 class (see MALDIquant R package).
absx	vector indicating lower and upper bounds for the mass-over-charge values to plot.
Peaks	MassPeaks object (see MALDIquant R package). If NULL, peaks are not highlighted.
Peaks2	numeric vector of mass-over-charge values to plot on the mass spectrum.
col_spec	color of the mass spectrum.
col_peak	color of the peak points corresponding to Peaks.
shape_peak	shape of the peak points corresponding to Peaks.
col_peak2	color of the peak points corresponding to Peaks2.
shape_peak2	Shape of the peak points corresponding to Peaks2.

Value

A ggplot object (see ggplot2 R package). Mass-over-charge values are in x-axis and intensities in y-axis.

Examples

```
library("MSclassifR")

# Load mass spectra
data("CitrobacterRKIspectra", package = "MSclassifR")
# Plot raw mass spectrum
PlotSpectra(SpectralData = CitrobacterRKIspectra[[1]])
# standard pre-processing of mass spectra
spectra <- SignalProcessing(CitrobacterRKIspectra)
# Plot pre-processed mass spectrum
PlotSpectra(SpectralData=spectra[[1]])
# detection of peaks in pre-processed mass spectra
peaks <- PeakDetection(x = spectra, labels = CitrobacterRKImetadata$Strain_name_spot)
# Plot peaks on pre-processed mass spectrum
PlotSpectra(SpectralData=spectra[[1]],Peaks=peaks[[1]],col_spec="blue",col_peak="black")
```

Predict_LogReg *Prediction of the category to which a mass spectrum belongs from a multinomial logistic regression model*

Description

This function predicts the category (species, phenotypes...) to which a mass spectrum belongs from a set of shortlisted mass-over-charge values of interest and a short-listed multinomial logistic regression model (see LogReg).

Usage

```
Predict_LogReg(peaks,model,moz,tolerance=6,normalizeFun=TRUE,noMatch=0)
```

Arguments

peaks	a list of MassPeaks objects (see MALDIquant R package).
model	a model or a list of models estimated from a set of shortlisted mass-over-charge values (output of the LogReg function).
moz	a vector with the set of shortlisted mass-over-charge values used to estimate the model Model.
tolerance	a numeric value of accepted tolerance to match peaks to the set of shortlisted mass-over-charge values. It is fixed to 6 m/z by default.
normalizeFun	a logical value, if TRUE (default) the maximum intensity will be equal to 1, the other intensities will be expressed in ratio to this maximum.
noMatch	a numeric value used to replace intensity values if there is no match detected between peaks and the set of shortlisted mass-over-charge values moz. It is fixed to 0 by default.

Value

Returns a dataframe containing probabilities of membership by category for each mass spectrum in peaks. The method used is provided in the method column. The comb_fisher method is the result of the Fisher's method when merging probabilities of membership of used prediction models.The max_vote method is the result of the maximum voting from used prediction models.

References

Kuhn, M. (2008). Building predictive models in R using the caret package. Journal of statistical software, 28(1), 1-26.

Examples

```
library("MSclassifR")
library("MALDIquant")
library("mixOmics")

####################################
## 1. Pre-processing of mass spectra

# load mass spectra and their metadata
data("CitrobacterRKIspectra","CitrobacterRKImetadata", package = "MSclassifR")
# standard pre-processing of mass spectra
spectra <- SignalProcessing(CitrobacterRKIspectra)
# detection of peaks in pre-processed mass spectra
peaks <- PeakDetection(x = spectra, labels = CitrobacterRKImetadata$Number_strain)
# matrix with intensities of peaks arranged in rows (each column is a mass-over-charge value)
```

```
IntMat <- MALDIquant::intensityMatrix(peaks)
rownames(IntMat) <- paste(CitrobacterRKImetadata$Strain_name_spot)
# replace missing values with 0 in the matrix
IntMat[is.na(IntMat)] <- 0
# normalize peaks according to the maximum intensity value for each mass spectrum
IntMat <- apply(IntMat,1,function(x) x/(max(x)))
# transpose the matrix for statistical analysis
X <- t(IntMat)
# define the known categories of mass spectra for the classification
Y <- factor(CitrobacterRKImetadata$Species)

######################################################################
## 2. Selection of discriminant mass-over-charge values using sPLS-DA

a <- SelectionVar(X,
                  Y,
                  MethodSelection = c("RFERF"),
                  MethodValidation = c("cv"),
                  PreProcessing = c("center","scale","nzv","corr"),
                  NumberCV = 2,
                  Sizes = c(5:10))
sel_moz=a$sel_moz

######################################################################
## 3. Perform LogReg from shortlisted discriminant mass-over-charge values
# and predict category of a mass spectrum

# Linear multinomial regression
model_lm=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2)

#nonlinear multinomial regression using neural networks
model_nn=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="nnet")
#Estimated model:
model_nn

#nonlinear multinomial regression using neural networks
model_rf=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="rf")
#Estimated model:
model_rf

#nonlinear multinomial regression using xgboost
model_xgb=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="xgb")
#Estimated model:
model_xgb

#nonlinear multinomial regression using svm
model_svm=MSclassifR::LogReg(X=X, moz=sel_moz, Y=factor(Y), number=2, repeats=2, kind="svm")
#Estimated model:
model_svm

#Of note, you can also load a model already saved (see example in LogReg function)

## Probabilities of belonging to each category for the mass spectra
```

```
prob_cat=MSclassifR::Predict_LogReg(peaks = peaks[c(5:7)],
model = list(model_lm$train_mod,model_nn$train_mod,model_rf$train_mod,
model_xgb$train_mod,model_svm$train_mod),
moz = sel_moz)
```

SelectionVar *Variable selection using random forests, logistic regression methods or sparse partial least squares discriminant analysis (sPLS-DA).*

Description

This function performs variable selection (i.e. selection of discriminant mass-over-charge values) using either recursive feature elimination (RFE) algorithm with Random Forest, or logistic regression model, or sparse partial least squares discriminant analysis (sPLS-DA).

Usage

```
SelectionVar(X,
             Y,
             MethodSelection = c("RFERF", "RFEGlmnet", "VSURF", "sPLSDA"),
             MethodValidation = c("cv", "repeatedcv", "LOOCV"),
             PreProcessing = c("center","scale","nzv","corr"),
             NumberCV = NULL,
             RepeatsCV = NULL,
             Sizes,
             Ntree = 1000,
             threshold = 0.01,
             ncomp.max = 10
             )
```

Arguments

X	a numeric matrix corresponding to a library of mass spectra. Each row of X is the intensities of a mass spectrum measured on mass-over-charge values.
Y	a factor with a length equal to the number of rows in X and containing the categories of each mass spectrum in X.
MethodSelection	a charater indicating the method used for variables selection. Four methods are avaible: with recursive feature elimination (RFE) and random forest ("RFERF"); logistic regression method ("RFEGlmnet"); method with random forest ("VSURF") and sparse partial least squares discriminant analysis ("sPLSDA").
MethodValidation	a charater indicating the resampling method:"cv" for cross-validation; "repeatedcv" for repeated cross-validation; and "LOOCV" for leave-one-out cross-validation.

NumberCV	a numeric value indicating K-folds for cross-validation. Don't used for "VSURF" method.
RepeatsCV	a numeric value indication the number of repeat(s) for K-folds for cross-validation or repeated cross-validation. Don't used for "VSURF" method.
PreProcessing	a vector indicating the method(s) used to pre-process the mass spectra in X: centering ("center"), scaling ("scale"), eliminating near zero variance predictors ("nzv"), or correlated predictors ("corr").
Sizes	a numeric vector indicating the number of variables to select. Don't used for "VSURF" method.
Ntree	a numeric value indicating the number of trees in each forest, only used if MethodSelection = "VSURF" (1000 by default).
ncomp.max	a positive Integer indicating the maximum number of components included in the sPLS-DA model (10 by default).
threshold	a positive Integer corresponding to a threshold used for optimal selection of the number of components included in the sPLS-DA model (0.01 by default).

Details

See rfe in the caret R package, VSURF in the VSURF R package and splsda in the mixOmics R package for details.

Value

A list composed of:

sel_moz	a vector with discriminant mass-over-chage values.

And of the results of the rfe function of the caret R package (methods RFERF and RFEGlmnet), or of the VSURF function of the VSURF R package (method VSURF).

For the sPLSDA method, it also returns the following items:

Raw_data	a horizontal bar plot and containing the contribution of features on each component.
selected_variables	data frame with uniques features (selected variables to keep and containing the contribution of features in order to class samples).See plotLoadings in the mixOmics R package for details.

References

Kuhn, Max. (2012). The caret Package. Journal of Statistical Software. 28.

Genuer, Robin, Jean-Michel Poggi and Christine Tuleau-Malot. VSURF : An R Package for Variable Selection Using Random Forests. R J. 7 (2015): 19.

Friedman J, Hastie T, Tibshirani R (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22.

Kim-Anh Le Cao, Florian Rohart, Ignacio Gonzalez, Sebastien Dejean with key contributors Benoit Gautier, Francois, Bartolo, contributions from Pierre Monget, Jeff Coquery, FangZou Yao and Benoit Liquet. (2016). mixOmics: Omics. Data Integration Project. R package version 6.1.1. https://CRAN.R-project.org/package=mixOmics

See Also

Vignettes MSclassifR : https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Ecrobia.html https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Klebsiella.html

Examples

```
library("MSclassifR")
library("MALDIquant")

####################################
## 1. Pre-processing of mass spectra

# load mass spectra and their metadata
data("CitrobacterRKIspectra","CitrobacterRKImetadata", package = "MSclassifR")
# standard pre-processing of mass spectra
spectra <- MSclassifR::SignalProcessing(CitrobacterRKIspectra)
# detection of peaks in pre-processed mass spectra
peaks <- MSclassifR::PeakDetection(x = spectra, labels = CitrobacterRKImetadata$Strain_name_spot)
# matrix with intensities of peaks arranged in rows (each column is a mass-over-charge value)
IntMat <- MALDIquant::intensityMatrix(peaks)
rownames(IntMat) <- paste(CitrobacterRKImetadata$Strain_name_spot)
# remove missing values in the matrix
IntMat[is.na(IntMat)] <- 0
# normalize peaks according to the maximum intensity value for each mass spectrum
IntMat <- apply(IntMat,1,function(x) x/(max(x)))
# transpose the matrix for statistical analysis
X <- t(IntMat)
# define the known categories of mass spectra for the classification
Y <- factor(CitrobacterRKImetadata$Species)


####################################
## 2. Perform variables selection using SelectionVar with RFE and random forest
## (with 5 to 10 variables)
a <- SelectionVar(X,
                  Y,
                  MethodSelection = c("RFERF"),
                  MethodValidation = c("cv"),
                  PreProcessing = c("center","scale","nzv","corr"),
                  NumberCV = 2,
                  Sizes = c(5:10))

# Plotting peaks on the first pre-processed mass spectrum and highlighting the
# discriminant mass-over-charge values with red lines
PlotSpectra(SpectralData=spectra[[1]],Peaks=peaks[[1]],
Peaks2=a$sel_moz,col_spec="blue",col_peak="black")

## 3. Perform variables selection unsing SelectionVar with RFE and logistic
```

```
## regression (with 5 to 10 variables)
## It is recommended to have a large enough data set to use this method
UpFeatures <-caret::upSample(X,Y, list = TRUE)
b <- SelectionVar(UpFeatures$x,
                  UpFeatures$y,
                  MethodSelection = c("RFEGlmnet"),
                  MethodValidation = c("cv"),
                  PreProcessing = c("center","scale","nzv","corr"),
                  NumberCV = 2,
                  Sizes = c(5:10))

# Plotting peaks on the first pre-processed mass spectrum and highlighting the
# discriminant mass-over-charge values with red lines
PlotSpectra(SpectralData=spectra[[1]],Peaks=peaks[[1]],
Peaks2=b$sel_moz,col_spec="blue",col_peak="black")

## 4. Perform variables selection unsing sPLDA method (with 5 to 10 variables per components)
#c <- SelectionVar(X,
#                  Y,
#                  MethodSelection = c("sPLSDA"),
#                  MethodValidation = c("LOOCV"),
#                  PreProcessing = c("scale","nzv"),
#                  Sizes = c(5:10))

# Plotting peaks on the first pre-processed mass spectrum and highlighting the
# discriminant mass-over-charge values with red lines
#PlotSpectra(SpectralData=spectra[[1]],Peaks=peaks[[1]],
#Peaks2=c$sel_moz,col_spec="blue",col_peak="black")

## 5. Perform variables selection using SelectionVar with RFE and logistic
## regression (with 5 to 10 variables per components)
  ## This function can last a few minutes

  d <- SelectionVar(X, Y, MethodSelection = c("VSURF"))
  summary(d$result)
```

SignalProcessing *Function performing post acquisition signal processing*

Description

This function performs post acquisition signal processing for `list` of `MassSpectrum` objects using commonly used methods : transform intensities ("sqrt"), smoothing ("Wavelet"), remove baseline ("SNIP"), calibrate intensities ("TIC") and align spectra. Methods used are selected from the `MALDIquant` and `MALDIrppa` R packages.

Usage

```
SignalProcessing(x,
                 transformIntensity_method = "sqrt",
                 smoothing_method = "Wavelet",
                 removeBaseline_method = "SNIP",
                 removeBaseline_iterations = 25,
                 calibrateIntensity_method = "TIC",
                 alignSpectra_halfWs = 11,
                 alignSpectra_SN = 3,
                 tolerance_align = 0.002,
                 ...)
```

Arguments

x	a list of MassSpectrum objects (see MALDIquant R package).
transformIntensity_method	a character indicating the method used to transform intensities: "sqrt" by default.
smoothing_method	a character indicating the smoothing methods used. By default, it performs undecimated Wavelet transform (UDWT) for list of MassSpectrum objects. See wavSmoothing in the MALDIrppa R package for details.
removeBaseline_method	a character indicating the method used to remove baseline. It uses "SNIP" method for list of MassSpectrum objects. See removeBaseline-methods of the MALDIquant R package for details.
removeBaseline_iterations	a numeric value indicting the number of iterations to remove baseline (by default = 25). See removeBaseline-methods of the MALDIquant R package for details.
calibrateIntensity_method	a character indicating the intensities calibration method used ("TIC" method by default). See calibrateIntensity-methods of the MALDIquant R package for details.
alignSpectra_halfWs	a numeric value half window size to detect peaks (by default = 11). See detectPeaks-methods of the MALDIquant R package for details.
alignSpectra_SN	a numeric value indicating the signal-to-noise ratio used to detect peaks (by default = 3). See detectPeaks-methods of the MALDIquant R package for details.
tolerance_align	a numeric value indicating a maximal relative deviation of a peak position (mass) to be considered as identical (by default = 0.002). See determineWarpingFunctions of the MALDIquant R package for details.
...	other arguments from MALDIquant and MALDIrppa packages.

Details

The Wavelet method relies on the wavShrink function of the wmtsa package and its dependencies (now archived by CRAN). The original C code by William Constantine and Keith L. Davidson, in turn including copyrighted routines by Insightful Corp., has been revised and included into MALDIrppa for the method to work.

All the methods used for SpectralTreatment functions are selected from MALDIquant and MALDIrppa packages.

Value

A list of modified MassSpectrum objects (see MALDIquant R package) according to chosen arguments.

References

Gibb S, Strimmer K. MALDIquant: a versatile R package for the analysis of mass spectrometry data. Bioinformatics. 2012 Sep 1;28(17):2270-1. doi: 10.1093/bioinformatics/bts447. Epub 2012 Jul 12. PMID: 22796955.

Javier Palarea-Albaladejo, Kevin Mclean, Frank Wright, David G E Smith, MALDIrppa: quality control and robust analysis for mass spectrometry data, Bioinformatics, Volume 34, Issue 3, 01 February 2018, Pages 522 - 523, doi: 10.1093/bioinformatics/btx628

See Also

Vignettes MSclassifR : https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Ecrobia.html https://agodmer.github.io/MSclassifR_examples/Vignettes/Vignettemsclassifr_Klebsiella.html

Examples

```
library("MALDIquant")
library("MSclassifR")

## Load mass spectra
data("CitrobacterRKIspectra", package = "MSclassifR")

# plot first unprocessed mass spectrum
PlotSpectra(SpectralData=CitrobacterRKIspectra[[1]], col_spec="blue")

## spectral treatment
spectra <- SignalProcessing(CitrobacterRKIspectra,
                            transformIntensity_method = "sqrt",
                            smoothing_method = "Wavelet",
                            removeBaseline_method = "SNIP",
                            removeBaseline_iterations = 25,
                            calibrateIntensity_method = "TIC",
                            alignSpectra_halfWs = 11,
                            alignSpectra_SN = 3,
                            tolerance_align = 0.002)
```

```
# plot first processed mass spectrum
PlotSpectra(SpectralData=spectra[[1]], col_spec="blue")
```

Index